U rendezés

Tekintsük a következő 14 elemű számsorozatot, és tegyük növekvő sorrendbe a leírás alapján!

Index	0	1	2	3	4	5	6	7	8	9	10	11	12	13
Elem	18	23	17	19	25	28	16	21	16	15	23	18	19	20

A rendezés gondlatmenete a következő:

- Menjünk végig az adatsorozaton az első elemtől az utolsó előtti elemig - az elem, ahol tartunk legyen indexelve az **akt** számmal.
- Minden **akt** érték mellett nézzük meg, hogy az aktuális szám nagyobb-e az őt közvetlenül követő számnál. Ha igen, akkor cseréljük fel őket.
- Egy ilyen bejárás után láthatjuk, hogy a kisebb elemek a sorozat eleje felé mentek, a nagyobb elemek a sorozat vége felé. Az is észrevehető, hogy a legnagyobb szám a sorozat legvégén van.

A sorozat számainak változását lépésről-lépésre mutatja az alábbi ábra (akt az első színes mező)! Figyeljük meg, hogy az előző állapothoz képest mikor volt csere!

Index	0	1	2	3	4	5	6	7	8	9	10	11	12	13
Elem	18	23	17	19	25	28	16	21	16	15	23	18	19	20

Index	0	1	2	3	4	5	6	7	8	9	10	11	12	13
Elem	18	17	23	19	25	28	16	21	16	15	23	18	19	20

Index	0	1	2	3	4	5	6	7	8	9	10	11	12	13
Elem	18	17	19	23	25	28	16	21	16	15	23	18	19	20

Index	0	1	2	3	4	5	6	7	8	9	10	11	12	13
Elem	18	17	19	23	25	28	16	21	16	15	23	18	19	20

Index	0	1	2	3	4	5	6	7	8	9	10	11	12	13
Elem	18	17	19	23	25	28	16	21	16	15	23	18	19	20

Index	0	1	2	3	4	5	6	7	8	9	10	11	12	13
Elem	18	17	19	23	25	16	28	21	16	15	23	18	19	20

Index	0	1	2	3	4	5	6	7	8	9	10	11	12	13
Elem	18	17	19	23	25	16	21	28	16	15	23	18	19	20

Index	0	1	2	3	4	5	6	7	8	9	10	11	12	13
Elem	18	17	19	23	25	16	21	16	28	15	23	18	19	20

Index	0	1	2	3	4	5	6	7	8	9	10	11	12	13
Elem	18	17	19	23	25	16	21	16	15	28	23	18	19	20

Index	0	1	2	3	4	5	6	7	8	9	10	11	12	13
Elem	18	17	19	23	25	16	21	16	15	23	28	18	19	20

Index	0	1	2	3	4	5	6	7	8	9	10	11	12	13
Elem	18	17	19	23	25	16	21	16	15	23	18	28	19	20

Index	0	1	2	3	4	5	6	7	8	9	10	11	12	13
Elem	18	17	19	23	25	16	21	16	15	23	18	19	28	20

Index	0	1	2	3	4	5	6	7	8	9	10	11	12	13
Elem	18	17	19	23	25	16	21	16	15	23	18	19	20	**28**

Láthatjuk, hogy a legnagyobb szám valóban a sorozat legvégére került.

Folytassuk úgy, hogy most **akt** értéke megint az első elemtől megy, de már nem kell elmennie az utolsó előtti elemig, hanem elég előbb befejeznie a munkát, tehát a példában a 11-es indexű elemig. Ha ezt a ciklust is végig visszük, akkor a sorozat így fog kinézni:

Index	0	1	2	3	4	5	6	7	8	9	10	11	12	13
Elem	17	18	19	23	16	21	16	15	23	18	19	20	**25**	**28**

Most látható, hogy a második legnagyobb elem az utolsó előtti helyre jutott. Így a következő végifutást elég eggyel rövidebb sorozaton végezni, tehát **akt** értéke most az elejétől csak a 10-es indexű elemig kell fusson. Ezután a sorozat tartalma a következő lesz:

Index	0	1	2	3	4	5	6	7	8	9	10	11	12	13
Elem	17	18	19	16	21	16	15	23	18	19	20	**23**	**25**	**28**

A gondolatmenet folytatható, az utolsó körben **akt** értének csak 0-nak kell lennie, és össze kell vetnie az első és a második számot, hogy azok jó sorrendben legyenek, mert egyébként minden más elem már jó helyen van.

Az algoritmus tehát két részből áll: van egy **vege** mutató, ami először az utolsó elemre mutat, aztán az utolsó előttire, majd az ez előttire, és így tovább, végül a második számra. Minden **vege** érték mellett az **akt** mutató végigmegy a sorozat **vege** előtti részén, tehát az első elemtől a vege-1 indexű elemig, és megcseréli az **akt** indexű és az azt követő elemet, ha nincsenek jó sorrendben.

1. Nyisd meg az órán elkészített Rendezések programot!
2. Készíts egy **URend**() nevű eljárást, amiben megvalósítod a fent leírt rendezést!
3. A **URend**() függvény az órán elkészített rendezéshez hasonlóan adja vissza az összehasonlítások és a cserék számát!
4. A főprogramban hívd meg a most elkészített rendezést egy 30 hosszú számsorozattal, majd ír ki a rendezetlen és a rendezett sorozatot, valamint az összehasonlítások és a cserék számát!
5. A kész munkát **git add . // git commit –m “RendK” // git push** paranccsal töltsd föl a repóba!